CS 118 Computer Network Fundamentals

Project 2: Simple Window-based Reliable Data Transfer in C/C++

Roger Lee 704018489

SEASnet login: rogerl

Hannah Chou 004016361

SEASnet login: jaling

1. **Implementation Description**

**Protocol:**

In this project, we used UDP and Go-Back-N to transfer files from a server to client.

**Header Formats/Messages:**

The following are the message formats assuming that the program is functioning correctly.

DATA requested seq#0 ACK#0 FIN 0 content-length: <length>

The length of the request packet is the length of the requested file name plus one for the null byte.

DATA sent seq#<seq> ACK#<ack> FIN <0 or 1> content-length: <length>

The server sends out this message for each data packet that is sent to the client. FIN is 1 when the file has finished transferring and the server wants to indicate that the file transfer is complete, thus initiating the process for connection termination.

DATA received seq#<seq> ACK#<ack> FIN 0 content-length: <length>

This is the message output by the client or server when it gets a data packet. For the server, it is output when it receives a file transfer request. For the client, it indicates that a packet has been received and it should send an ACK back to the server.

ACK sent seq#<seq> ACK#<ack> FIN 0 content-length: <length>

This message is sent by the client when it receives a data packet from the server without any issues. When the packet is corrupted, however, the ACK is just not incremented to indicate to the server that the packet needs to be resent.

ACK received seq#<seq> ACK#<ack> FIN 0 content-length: <length>

This message indicates that the server has received the ACK from the client.

FINACK sent seq#<seq> ACK#<ack> FIN 1 content-length: <length>

The FINACK should always have the fin = 1. After the server sends the last packet of the file, it sends another packet to indicate that the file transfer is complete. After the server sends the data packet indicating that file transfer is complete, the client sends a FINACK with the above message back to the server acknowledge. The server then sends a last FINACK to the client, totaling two FINACKs in the connection termination process.

FINACK received seq#<seq> ACK#<ack> FIN 1 content-length: <length>

The FINACK should always have the fin = 1. After the server sends a FINACK, the client sends a FINACK back to acknowledge that the file transfer is complete. The message above is output when the server receives the clients FINACK. The connection is then terminated after the server sends a last FINACK to the client.

**Timeouts:**

We chose to have the timeout time be 2 seconds. Although this is a little slow in terms of typical file transfer, the delay made it easier to identify bugs when we were testing for dropped or corrupted packets.

1. **Difficulties and Challenges**

The biggest difficulty with this project was just keeping track of the numbers and iterations. Sometimes it was hard to tell whether the issue was on the client side or on the server side. After some debugging, we would realize there was some conditional statement that did not handle all the cases well. For example, at one point, the client would receive corrupted data packets but the server would go ahead and initiate termination of the connection. It turned out that the server would continue with sending the FIN packet right after the last packet has been sent. In other test cases with non-1 probabilities for corruption, this was not apparent because the last packet never had problems. However, when we sent the corruption probability to 1, we realized that there needed to be a check that the last packet was received correctly before the server could send a FIN packet.

Another difficulty we had with this project was using the functions sendto() and recvfrom(). It took a little while to figure out the issues preventing the packets from being transferred. It turns out the issue was that we had created a pointer to a pointer, instead of passing in a pointer to the packet struct as intended. It seemed like a minor detail but we had to fix it in several locations in our code. Once we got these to work, we could confirm that there was connection and continue with the project.